i, Poet: Automatic Chinese Poetry Composition through a Generative Summarization Framework under Constrained Optimization

i, Poet: Automatic Chinese Poetry Composition through a Generative Summarization Framework under Constrained Optimization

Rui Yan[†,♮], **Han Jiang**[♮], **Mirella Lapata**[‡], **Shou-De Lin**[†], **Xueqiang Lv**[♭], **and Xiaoming Li**[♮]
[†]Dept. of Computer Science and Information Engineering, National Taiwan University, Taiwan
[‡]Institute for Language, Cognition and Computation, University of Edinburgh, U.K.
[♮]Dept. of Computer Science, Peking University, China
[♭]Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, BISTU, China
{*r.yan, billybob, lxq, lxm*}*@pku.edu.cn, mlap@inf.ed.ac.uk, sdlin@csie.ntu.edu.tw*

Abstract

Part of the long lasting cultural heritage of China is the classical ancient Chinese poems which follow strict formats and complicated linguistic rules. Automatic Chinese poetry composition by programs is considered as a challenging problem in computational linguistics and requires high Artificial Intelligence assistance, and has not been well addressed. In this paper, we formulate the poetry composition task as an optimization problem based on a generative summarization framework under several constraints. Given the user specified writing intents, the system retrieves candidate terms out of a large poem corpus, and then orders these terms to fit into poetry formats, satisfying tonal and rhythm requirements. The optimization process under constraints is conducted via iterative term substitutions till convergence, and outputs the subset with the highest utility as the generated poem. For experiments, we perform generation on large datasets of 61,960 classic poems from *Tang* and *Song* Dynasty of China. A comprehensive evaluation, using both human judgments and ROUGE scores, has demonstrated the effectiveness of our proposed approach.

1 Introduction

The ancient Chinese classical poetry is a special and important cultural heritage with more than thousands of years in history. As opposed to modern poetry, the classic poems have unique elegance, e.g., aestheticism and conciseness etc.

Composing classic poems is considered as a challenging task and only few modern people can master such a skill: it is hard either to manipulate or to organize terms. We realize that computers might play an important role in helping humans to create classic poems: 1) it is rather convenient for computers to sort out appropriate term combinations from a large corpus, and 2) creating classic poetry requires people to follow many rules and patterns, which can naturally be written as constraints to optimize by machines: the advantages motivate automatic poetry generation via computational intelligence.

江雪
(*Snow, River*)
千山鸟飞绝，
(*All birds have hidden.*)
万径人踪灭。
(*All people have disappeared.*)
孤舟蓑笠翁，
(*A man in a straw hat sits on a lonely boat,*)
独钓寒江雪。
(*fishing in the snow on the river.*)

Table 1: One example of the manually generated poem. Particularly, the rhyming characters are shown in red color.

For people to better inherit this classic art, we introduce a meaningful task of automatic Chinese poetry composition, aiming to endow the computer with artificial intelligence to mimic the generation process of human poetry so that it would be a tool that aids people to master proficiency in poem composition. We name the system as **iPoet**, which indicates our goal is that everyone could announce proudly "I, a poet".

To design automatic poetry composition, we first need to study its generation criteria. There are several writing formats for Chinese poetry, while *quatrain* (namely 绝句, consisting of 4 lines of sentences) and *regulated verse* (namely 律诗, 8 lines of sentences), either with 5 or 7 characters per sentence, show dominative culture prominence throughout the Chinese history. In this paper, we mainly focus on the generation of quatrains, but the idea can also be applied to the generation of regulated verses as both processes are theoretically similar.

Unlike narratives, which follow less strict rules and restrictions, a classical poem has rigid tonal pattern, rhyme scheme and structural constraints. We illustrate a sample poem written by the famous poet named *Zongyuan Liu* in Table 1, from which we can see that there are several prominent attributes which serve as standards for poetry composition.

• *Structure.* The number of characters (or syllables) of all

lines ought to be uniform, with 5 or 7-char in a line; no irregular line is allowed. Therefore, the total number of characters for a poem is then fixed: 20 or 28 for a 4-line quatrain.

• *Rhyming.* In Chinese poetry, rhyming characters share the same ending vowel, and the ending characters in most of the sentences within a poem need to be the same rhyme, so that the poem sounds harmonious. For instance, the rhyming constraints for a Chinese quatrain are:

1) The ending characters of the 1st, 2nd and 4th sentences should rhyme, as the red characters in Table 1. The ending vowel of the 3rd sentence does not necessarily rhyme.

2) There shall be no replicated characters to rhyme within a single poem, i.e., no rhyming character should be used twice.

• *Tonal.* In traditional Chinese, every character consists of one syllable and has one tone which can be classified as either *level-tone* (平) or *downward-tone* (仄). The general principle for the tonal pattern in a classical poem is that it is preferable to have these two kinds of tones interleaved in every sentence to avoid monotone.

• *Semantic Coherence.* Finally, as is generally true for all meaningful writings, a well-written poem is supposed to be semantically coherent among all lines. One way to guarantee this is to ensure the terms share high contextual coherence.

We propose the iPoet system in a generative summarization framework under constrained optimization: the mentioned poetry criteria are formulated as mathematical constraints to refine the poetry generation. Taking keywords from users as the description of their intent, iPoet selects term subsets out of all candidates to generate the lines by iterative term substitution via a *ranking and re-ranking* process.

A generative summarization style for specific poetry composition is obviously different from the traditional standard summarization. Our contributions are as follows:

• As there is no existing general method to compose Chinese poems, it is challenging to model this important problem. Our 1st contribution is to formulate the problem as a specific generative summarization framework under linguistic constraints, which is a novel insight.

• We have several criteria for poetry composition, and it is challenging to formulate the rules as functions and constraints. The 2nd contribution is to incorporate poetic characteristics into the generative summarization framework under constraint optimization, which is never studied before.

We build iPoet on the datasets of 61,960 poems to verify its effectiveness compared with 5 baselines, and illustrate some interesting examples generated by iPoet in the later section. We start by reviewing previous works. In Section 3 & 4 we formulate a generative summarization framework via iterative substitution, and propose several constraints. We describe experiments in Section 5, and draw conclusions in Section 6.

2 Related Work

As poetry is one of the most significant literature heritage of various cultures all over the world, there are some formal researches into the area of computer-assisted poetry generation. Scientists from different countries have studied the automatic poem composition in their own languages through different manners: 1) Genetic Algorithms. Manurung *et al.* propose to create poetic texts in English based on state search [2004; 2011]; 2) Statistical Machine Translation (SMT). Greene *et al.* propose a translation model to generation cross-lingual poetry, from Italian to English [2010]; 3) Rule-based Templates. Oliveira has proposed a system of poem generation platform based on semantic and grammar templates in Spanish [2009; 2012]. An interactive system has been proposed to reproduce the traditional Japanese poem named *Haiku* based on rule-based phrase search related to user queries [Tosa *et al.*, 2008; Wu *et al.*, 2009]. Netzer *et al.* propose another way of Haiku generation using word association rules [2009].

Besides studies in English, Japanese, Spanish and Italian poetry composition, there is continuing research on Chinese poetry as well, because Chinese poetry has its intrinsically special rules to follow. Most of the approaches proposed for other languages might not be directly applicable.

There are now several Chinese poetry generators available, usually template based. The system named *Daoxiang*[1] basically relies on manual pattern selection. The system maintains a list of manually created terms related to pre-defined keywords, and inserts terms *randomly* into the selected template as a poem. The system is simple but random term selection leads to unnatural sentences and incoherent contents.

Zhou *et al.* use a genetic algorithm for Chinese poetry generation by tonal-coding state search only [2010]. In a study of Chinese couplet generation, which could be narrowed down as a minimal poem form of 2 lines, a SMT model is proposed to generate the 2nd sentence given the 1st sentence of a couplet [Jiang and Zhou, 2008]. He *et al.* extend the SMT framework to generate a 4-line poem by translating previous sentences sequentially, considering structural templates [2012].

However, none of these methods considers overall dependency of semantic coherence of the composed poems. To the best of our knowledge, we are the first to apply the generative summarization framework for the poetry generation problem: the summarization is *generative* rather than *extractive* since we need to generate sentences by re-organizing terms. The proposed iterative term substitution strategy meets sophisticated poem criteria including semantic coherence.

3 System Overview

One plausible procedure for a poet to create a poem is to first outline the main writing intents, which could be represented by a set of keywords. Then the author chooses a particular format, e.g., the quatrain, creating and ordering a number of relevant terms to form the poem which satisfies the tone and rhyme [Wang, 2002]. It is an iterative process since the author can always change part of terms to polish the idea till the entire poem is finished. iPoet tries to imitate such a process.

We define the problem formulation as follows:

Input. Given the keywords of κ=$\{\kappa_1, \kappa_2, \ldots, \kappa_{|\kappa|}\}$ from an author as the writing intent (i.e., topic, subject, or theme for the poem to generate), where κ_i is a keyword term, we obtain a candidate term set Ω=$\{w|w \in \Omega\}$ from our corpus.

Output. We generate a poem $\mathcal{P}$=$\{w|w \in \mathcal{P}\}$, where the term w is selected from set Ω to fit the poetry format ($\mathcal{P} \subseteq \Omega$).

[1] http://www.poeming.com/web/index.htm

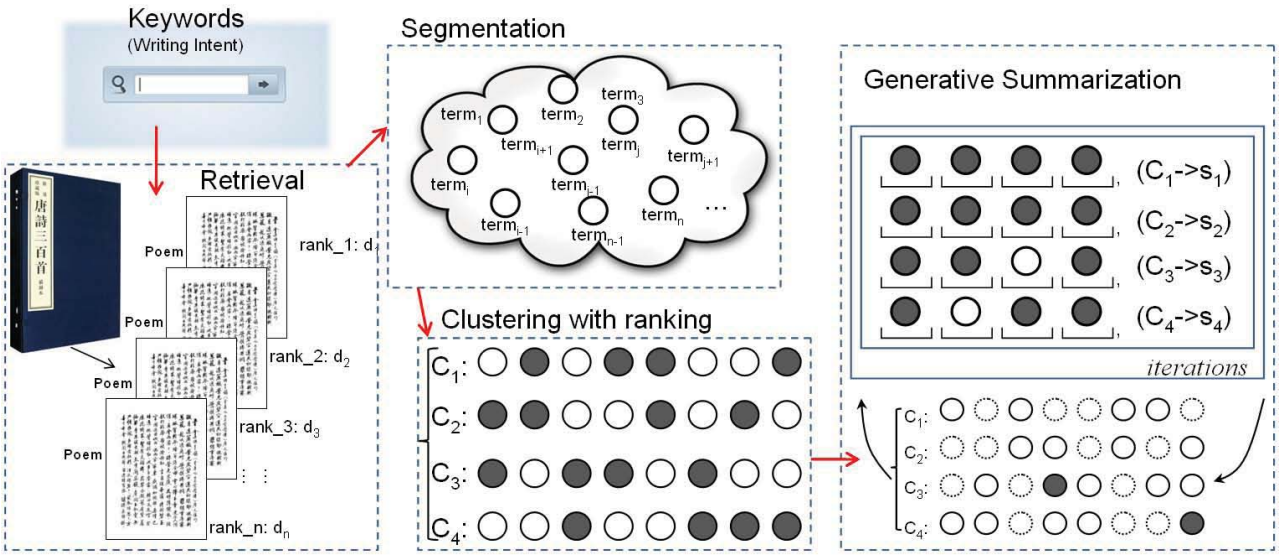

Figure 1: Illustration of the iPoet system framework. The system takes the users' writing intents as queries, and retrieves poems out of the collections. After term segmentation and clustering, the poem is composed in a generative summarization manner. The white circles denote all candidate terms while the shaded ones indicate the chosen terms after iterative substitutions.

System Framework. The system accepts the user's keywords κ as a "query", where κ represents the overall subject of the poem. Then it seeks out and ranks all query related poems as a candidate corpus. The process is basically an Information Retrieval procedure. Given the retrieved poems, we apply segmentation techniques to truncate the poems into terms and characters, applying the toolkit[2] to annotate syllables according to the tonal dictionary of Chinese characters. The system then groups these terms into different semantic clusters with rankings, and finally generate every line of the poem out of each cluster in a generative summarization manner, selecting terms with poetic preference and constraints to construct a sentence. The composition process is shown in Figure 1 and the details will be discussed in the next section.

4 Automatic Chinese Poetry Composition

In this section, we will introduce components of the iPoet system step by step, including: 1) *retrieval*, 2) *segmentation*, 3) *clustering* and 4) *generation*, while the former 3 parts could be described as *Pre-Generation Process* as a whole.

4.1 Pre-Generation Process

After taking in the user issued writing intents κ which could be one or more terms, we apply a standard retrieval process via keyword search on the poem datasets, using the open source platform Lucene[3]. The poem corpus are indexed in an inverted index prepared offline. This retrieval process returns a ranked list of relevant documents of poems $\mathcal{D}^\kappa$=$\{d|d_1^\kappa, d_2^\kappa, \ldots, d_{|\mathcal{D}^\kappa|}^\kappa\}$ from the poem corpus $\mathcal{D}$, where d is a poem with a ranking score $\psi(d^\kappa)$ which denotes the score of d in correspondence to keywords κ.

[2]http://cls.hs.yzu.edu.tw/makepoem/home.htm
[3]http://lucene.apache.org/

These retrieved documents are then segmented into terms. Each term w is initiated with a weight from two aspects:

Relevance. Given keywords κ, every retrieved document is associated with a ranking score of relevance $\psi(d^\kappa)$ during the retrieval process. For brevity, we remove the superscript κ when there is no ambiguity. Intuitively, the terms with high scores have a larger chance to be included in the poem.

Importance. We evaluate the importance of the terms within each poem d according to the standard *tf-idf* weighting, which reflects how important a term is within the document [Neto *et al.*, 2000]. $\pi_0(w, d)$ denotes the initial importance of w measured in d, while the weight of d incorporated.

$$\pi_0(w,d) = \frac{tf(w,d)(1+\log\frac{|\mathcal{D}|}{N_w})}{\sqrt{\sum_{w'\in d}(tf(w',d)(1+\log\frac{|\mathcal{D}|}{N_{w'}}))^2}} \quad (1)$$

$tf(w,d)$ is the term frequency in d and $\log\frac{|\mathcal{D}|}{N_w}$ is the inverse document frequency where N_w is the number of documents containing w. The overall initial weight of w denoted as $\pi_0(w)$ is calculated as a weighted linear summation of the collection so that the initial weight of every candidate is:

$$\pi_0(w) = \sum_{i=1}^{|\mathcal{D}|} \psi(d_i) \cdot \pi_0(w, d_i) \quad (2)$$

As *topics* have long been investigated as the significant latent aspects of terms [Hofmann, 2001; Landauer *et al.*, 1998]. Here the Latent Dirichlet Allocation [Blei *et al.*, 2003] model is used to discover topics and to cluster terms. We also obtain the probability distribution over topics assigned to a term w, i.e., $p(w|z)$. The inferred topic representation is the probabilities of terms belonging to the topic z, which is

$$z = \{p(w_1|z), p(w_2|z), \ldots, p(w_i|z)\}$$

We train a *100*-topic model and represent each term using its invert hidden topic distribution obtained from LDA, shown

as follows, where each w is represented as a topic vector $\vec{w}$. We measure the term correlation on the dimension of all topics by the cosine similarity on topic vector. Intuitively, we tend to cluster together terms which are of similar subjects and each cluster of terms would be used to produce one single sentence. Below is the distance metric for clustering.

$$\phi(w, w') = \frac{\vec{w} \cdot \vec{w}'}{||\vec{w}||||\vec{w}'||} \tag{3}$$

where

$$\vec{w} = \{p(w|z_1), p(w|z_2), \ldots, p(w|z_{100})\}$$

4.2 Generative Summarization

After pre-generation, we obtain m term clusters, each denoted as C_i. To generate a quatrain, m = 4, and each cluster is responsible for generating a line in a quatrain. Within each cluster, terms are ranked by the initial weight $\pi_0(.)$.

Seeds Selection. Since rhythm is an essential element for poems, our generation process starts from capturing the rhythmical characters as seeds in the beginning.

We aim to choose seed terms with both high weights and the same vowel. The seeds are initiated as the highest ranked terms, one for each line/cluster. The most idealistic situation is for these terms to share the same vowel while most likely it is not this case. We hence traverse the candidate terms to find an optimal seed set S=$\{w_{s_i|=1\sim m}|w_{s_i} \in C_i\}$ to maximize the total weight, and w_s is a seed term. Function $v(c(w))$ is to examine the vowel of the last character $c(w)$ for the term w. As in Equation 4, we select m distinct seeds for each line.

$$\begin{aligned} S = &\underset{w_{s_i} \in C_i}{\text{argmax}} \sum\nolimits_{w_{s_i}} \pi_0(w_{s_i}) \quad i = 1, 2, \ldots, m \\ &\text{subject to: } \forall\, c(w_s) \neq c(w_{s'}), v(c(w_s)) = v(c(w_{s'})) \end{aligned} \tag{4}$$

Poetry Generation. We aim to select terms of large correlativeness among all generated lines, especially cross-lines namely the *cross-line coherence*, taking all lines into account simultaneously, which has never been considered before.

Coherence. Similar to the pilot studies on summarization coherence investigation in [Yan *et al.*, 2011a; 2011b], a poem consists of a series of individual but correlated lines. A well generated line should be semantically aligned with other lines to avoid undesired content drift or distraction in meaning. We model the coherence $\pi_c(.)$ for the term w as:

$$\pi_c(w) = \mu \cdot \sum\nolimits_{w' \in \mathcal{P}(w)} \phi(w, w') + (1-\mu) \cdot \sum\nolimits_{w'' \in \mathcal{P} \setminus \mathcal{P}(w)} \phi(w, w'') \tag{5}$$

$\mathcal{P}(w)$ is used to denote the line in $\mathcal{P}$ where w belongs to. The first component is an *intra*-line correlation while the other component is an *inter*-line correlation, controlled by $\mu \in [0,1]$. The refined significance score of $\pi(w)$ is calculated with a linear interpolation, considering *relevance*, *importance* and *coherence*. The parameter $\lambda \in [0, +\infty)$ is to control the combination of initial weights and weights via iteration:

$$\pi(w) = \pi_0(w) + \lambda \cdot \pi_c(w) \tag{6}$$

For the poem to compose, we aim to maximize the total weights of the chosen terms as the poem utility $\mathcal{U}(\mathcal{P})$, trying different candidate subsets. The task is to obtain the utility optimized term subset of $\mathcal{P}^\star$ from the space of all combinations of subsets. The preliminary objective function is as follows:

$$\begin{aligned} \mathcal{P}^\star &= \underset{\mathcal{P}}{\text{argmax}}\, \mathcal{U}(\mathcal{P}) \\ &= \underset{\mathcal{P}}{\text{argmax}} \sum\nolimits_{w \in \mathcal{P}} \pi(w) \end{aligned} \tag{7}$$

As the refined significance score $\pi(w)$ is measured by the neighboring lines in the generated poem in our framework, we generate $\mathcal{P}$ *iteratively* to approximate $\mathcal{P}^\star$, i.e., maximize total significance based on the candidate poem generated in the last iteration. Note that $\pi_c(.)$ in $\pi(.)$ is dynamic as the candidate term set is subject to change after each iteration.

Based on the term weight, we obtain a ranking list for each cluster C_i. Top ranked terms are of higher importance, relevance and coherence, and are strong candidates to be selected into the line of $\mathcal{P}_i$. Given a line $\mathcal{P}_i^{(n-1)}$ generated in the *(n-1)*-th iteration and the top ranked terms in the *n*-th iteration (denoted as $\mathcal{I}_i^{(n)}$), they have an intersection set of $\mathcal{Z}_i^{(n)}$=$\mathcal{P}_i^{(n-1)} \cap \mathcal{I}_i^{(n)}$. There is a substitutable term set $\mathcal{X}_i^{(n)}$=$\mathcal{P}_i^{(n-1)}$-$\mathcal{Z}_i^{(n)}$ and a new incoming candidate term set $\mathcal{Y}_i^{(n)}$=$\mathcal{I}_i^{(n)}$-$\mathcal{Z}_i^{(n)}$. Under defined constraints, we substitute $\mathrm{x}_i^{(n)}$ terms with $\mathrm{y}_i^{(n)}$, where $\mathrm{x}_i^{(n)} \subseteq \mathcal{X}_i^{(n)}$ and $\mathrm{y}_i^{(n)} \subseteq \mathcal{Y}_i^{(n)}$. During every iteration, our goal is to find a substitutive pair $<\mathrm{x}_i, \mathrm{y}_i>$ for $\mathcal{P}_i$. To measure the performance of such substitution, a discriminant gain function

$$\begin{aligned} \Delta\, \mathcal{U}^{(n)}_{\mathrm{x}_i^{(n)}, \mathrm{y}_i^{(n)}} &= \mathcal{U}(\mathcal{P}_i^{(n)}) - \mathcal{U}(\mathcal{P}_i^{(n-1)}) \\ &= \mathcal{U}((\mathcal{P}_i^{(n-1)} - \mathrm{x}_i^{(n)}) \cup \mathrm{y}_i^{(n)}) - \mathcal{U}(\mathcal{P}_i^{(n-1)}) \end{aligned} \tag{8}$$

is employed to quantify the penalty. Therefore, we predict the substitutive pair by maximizing the gain function $\Delta\mathcal{U}$. Note that all lines can be generated in parallel independently since they maintain different substitutable term pairs.

Recall the criteria mentioned for poetry generation. We opt to choose the weight-maximized substitutive pairs under these constraints as iterations accumulate:

1) *Structure.* The number of each line would not exceed a constant τ, and e.g., in our case, τ=5 for a 5-character quatrain. Hence, each line $\mathcal{P}_i$ in $\mathcal{P}$ has a constraint of $|\mathcal{P}_i| \leq \tau$.

2) *Rhyming.* This constraint is met by seed selection.

3) *Tonal.* We adjust the order of the selected terms within a line so that the tonal sequence will conform to one of the general tonal patterns (e.g., the one shown in Section 1), which could be denoted as $tonal(\mathcal{P}) \propto \mathbb{T}$, where $\mathbb{T}$ denotes a general tonal pattern, and '$\propto$' means 'conform to'. If no possible order of the selected term by a particular substitution can meet the tonal constraint, the substitutive pair would be rejected.

4) *Semantic Coherence.* The coherence is captured in term weighting and poem optimization during iteration.

Finally the objective function based on Equation (8) can be designed as a maximization of utility gain by substituting x with y during each iteration considering these constraints:

$$<\mathrm{x}_i, \mathrm{y}_i> = \underset{\mathrm{x}_i \subseteq \mathcal{X}_i, \mathrm{y}_i \subseteq \mathcal{Y}_i}{\text{argmax}} \Delta\mathcal{U}_{\mathrm{x}_i, \mathrm{y}_i}, \tag{9}$$

subject to:

(1) $|\mathcal{P}_i| \leq \tau$,
(2) $\forall\, c(w_s) \neq c(w_{s'}), v(c(w_s)) = v(c(w_{s'}))$,
(3) *tonal* $(\mathcal{P}) \propto \mathbb{T}$.

We rank all substitutable pairs by the utility gain, and a straight understanding is that we find a maximized utility gain $\Delta\mathcal{U}_{\mathtt{x}_i,\mathtt{y}_i}$ for each line $\mathcal{P}_i$, while at the same time the constraints are satisfied. We start from the best ranked substitutable pair in each line, and try to find one pair for each line among all lines through a process of *Dynamic Programming*. After applying the substitution of $\Delta\mathcal{U}_{\mathtt{x}_i,\mathtt{y}_i}$, a line $\mathcal{P}_i$ is hence generated within this iteration and the poem is created by choosing a "path" among all pair candidates.

In this way, all individual lines are updated after each iteration, depending on the other lines. The convergence of the algorithm is achieved when the utility gain drops under a threshold ϵ, which is set at 0.001 in this study.

5 Experiments and Evaluation

5.1 Data

During the *Tang Dynasty* (618-907 A.D.) and *Song Dynasty* (960-1279 A.D.), Chinese literature reached its golden age. We downloaded "*Poems of Tang Dynasty*" (PTD), "*Poems of Song Dynasty*" (PSD) from the Internet, which amounts to 61,960 poems. More detailed statistics are listed in Table 2, which shows the number of total lines, unique terms, unique characters in the corpus, and the average length per line.

Table 2: Detailed basic information of the poem datasets.

	#Poem	#Line	#Term	#Char.	#Avg.Len
PTD	42,974	463,825	7,513	10,205	5.58
PSD	18,986	268,341	5,750	6,996	4.88

5.2 Evaluation Metrics

As we use a generative summarization based method to compose poetry, we try to apply the evaluation metrics for summarization methods to judge the performance of poetry algorithms. Document Understanding Conference (DUC) officially employs ROUGE measures for document summarization performance evaluation, counting the number of overlapping units such as N-grams, word sequences, and word pairs between the candidates and the references [Lin and Hovy, 2003]. Several automatic evaluation methods are implemented in ROUGE, such as ROUGE-N, ROUGE-L and ROUGE-W. Take ROUGE-N as an example:

$$\text{ROUGE-N} = \frac{\sum_{\text{P}\in\text{RP}} \sum_{\text{N-gram}\in\text{P}} \text{Count}_{\text{match}}(\text{N-gram})}{\sum_{\text{P}\in\text{RP}} \sum_{\text{N-gram}\in\text{P}} \text{Count (N-gram)}}$$

N-gram$\in$RP denotes the N-grams in the reference poems. $\text{Count}_{\text{match}}$(N-gram) is the maximum number of N-gram in the generated poem and in reference poems. Count(N-gram) is the number of N-grams in the reference or generated poem.

Since we have similar result observations in any sub-metric of ROUGE scores, we only report ROUGE-1, ROUGE-2, and the weighted longest common subsequence based ROUGE-W (W=1.2) for comparison by applying ROUGE ver 1.55.

In our experiments, we use the top-10 retrieved poems by human poets as the reference poems to evaluate the generated poem by computer. To the best of our knowledge, there is no previous work about poetry evaluation with ROUGE metric. Intuitively, for ROUGE scores, the higher the better.

We also include human judgements from 25 evaluators who are graduate students majoring in Chinese literature. Evaluators are requested to express an opinion over the automatically composed poems. A clear criterion is necessary for human evaluation. We adopt the evaluation standards discussed in [Wang, 2002; He *et al.*, 2012]: "Fluency", "Rhyme", "Coherence", and "Meaning", which the human judges can easily follow. They need to assign binary scores for each of the four criteria ('0'-no, '1'- yes). After that, the total score of the poem is calculated by summing up the four individual scores, in a 5-point scale ranging from 0 to 4.

5.3 Comparison Algorithms

We implemented several intuitive poetry generation methods, and the following widely used algorithms for general summarization as baselines since our proposed method is also under the framework of summarization. For fairness, we conduct the same pre-generation process to all algorithms.

Random. Terms are randomly chosen to order as a poem.

ITW. Given the retrieved poems, ITW ranks the segmented terms according to their Initial Term Weighting (i.e., based on only relevance and importance scores mentioned in Section 4) and selects the highly ranked ones to fit into the poem.

Centroid. The method applies MEAD [Radev *et al.*, 2004] to extract centroid terms and terms near the centroid, following parameters of centroid value and positional value.

GBS. The idea of Graph-Based Summarization is to construct a term connectivity graph established by term correlativeness and select important terms to compose the poem based on eigenvector centrality [Wan and Yang, 2008].

SMT. A Chinese poetry generation method is proposed by Statistical Machine Translation [He *et al.*, 2012]. The process is that given one generated line, the system generates the next line by translating the previous line of sentence as a pair of "couplet" one by one, which is a single-pass generation.

iPoet. Our proposed poetry composition algorithm formulates the problem into a generative summarization framework as a multi-pass generation, taking importance, relevance and coherence under poetry-specific constraints into account.

5.4 Overall Performance Comparison

To narrow down the scope of feasible experiment testings, we limit the keywords that users may enter to describe their writing intents within the poetic phrase taxonomy [Liu, 1735]. In this taxonomy, 41,218 phrases (34,290 unique) of length ranging from 1 to 5 characters are classified into 1,016 categories which cover most common topics in poems [He *et al.*, 2012]. Each category is associated with a general concept.

We select the most representative phrases in the top-20 largest categories as our testing cases for evaluation and all the other categories as training. We report the average score in terms of ROUGE-*1*, ROUGE-*2*, ROUGE-*W* and *Human* (H) on all sets. Results are shown in Table 3 and Figure 2.

From the results, we have the following observations:

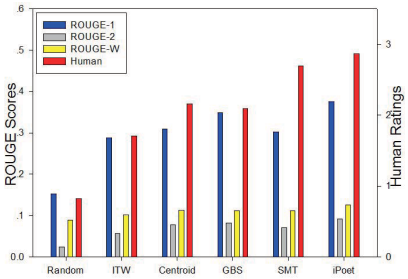

Figure 2: Overall performance on 2 datasets.

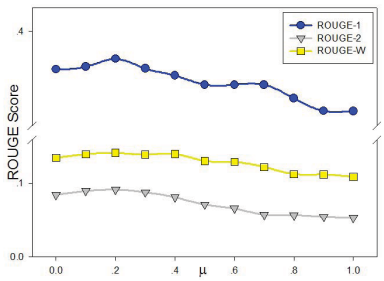

Figure 3: μ: the intra/inter-line coherence.

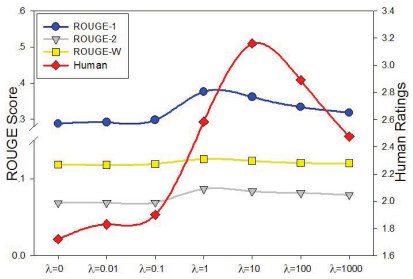

Figure 4: λ: control the weight of coherence.

Table 3: Overall performance comparison on the 2 datasets.

	Poem of Tang & Song Dynasty					
Algo.	Rand.	ITW	Cent.	GBS	SMT	iPoet
R-1	0.153	0.293	0.307	0.318	0.315	**0.385**
R-2	0.022	0.056	0.075	0.081	0.078	**0.086**
R-W	0.089	0.098	0.103	0.114	0.117	**0.129**
H	0.920	1.640	2.120	2.320	2.960	**3.160**

• The Random method has the worst performance as expected, since it is naive without considering any poem characteristics except for the general format.

• The ITW method generates poems choosing the most initially weighted terms. On average, its performance is not satisfying since no term correlation is considered.

• The results of Centroid are better than those of Random and ITW. This is because the Centroid algorithm selects the important terms (centroids) and the terms in near positions.

• GBS largely outperforms Centroid in ROUGE, which might be because the PageRank-based method by eigenvector centrality captures implicit coherence via term links. The performance is still relatively low by human judges.

• It is interesting that the translation based generation SMT achieves high scores in human evaluation but quite low performance in terms of ROUGE. It might be due to the fact that the generation process is restricted to the rigid sentence-to-sentence translation without coherence of all lines. Yet, the translation between sentences brings good user experience.

• No previous generation method considers the importance, relevance and full coherence simultaneously, especially the multi-pass refinement of all lines iteratively, while iPoet based on generative summarization easily captures these poem criteria. Our proposed framework could be naturally extended with more specific poetic constraints.

Having provided positive demonstrations and proved the effectiveness of iPoet, we move to parameter tuning and component analysis of *importance*, *relevance* and *coherence*.

5.5 Components and Parameters

We analyze the components of *relevance*, *importance* and *coherence* in isolation so as to assess their individual contributions, listed in Table 4. The first group of *Rel*, *Imp*, and *Coh* is performed using the corresponding components of relevance, importance and coherence only and the second group of *-Rel*, *-Imp*, and *-Coh* is performed using the full combination exempting the corresponding component (leave-one-out). From Table 4 we see that the component of *Coh* is a significant factor in poetry generation: we mark the most prominent change caused by components with asterisks.

Table 4: Examination of all components of the iPoet system.

	Imp.	Rel.	Coh.	-Imp.	-Rel.	-Coh.
R-1	0.249*	0.217	0.202	0.308	0.323	0.288*
R-2	0.058*	0.055	0.051	0.077	0.081	0.069*
R-W	0.103*	0.098	0.095	0.124	0.121	0.118*

There are two key parameter λ and μ to tune: λ indicates the relative weights of coherence to term weighting, and μ controls the tradeoff between *intra-/inter*-line coherence.

Through Figure 3 & 4, we see that when λ is small ($\lambda \in [0, 0.1]$), both human ratings and ROUGE scores have little difference. When $\lambda \in [1, 100]$, both scores increase significantly while user satisfaction reaches its peak around λ=10, which again indicates the effectiveness of coherence in poetry. The performance start to decay when λ grows larger. We scrutinize into the tradeoff between intra- and inter-line coherence by μ. The inter-line coherence is proved to be useful (μ=0.2).

6 Conclusion and Future Work

Poetry composition is a difficult task in the field of language generation. We propose a novel approach to model this problem into a generative summarization framework. Given the user writing intents as queries, we utilize the poetry corpus to generate Chinese poems, more specifically, quatrains. We formally formulate the summarization process via iterative term substitution. We apply a constrained optimization according to several poetry criteria, measured by *importance*, *relevance* and *coherence*, which is a multi-pass generation.

A series of experiments are done on datasets of *Tang* and *Song* Dynasty. We compare our approach with 5 baselines. For the first time, we apply ROUGE to evaluate the performance of poetry generation as well as human judgements. Through our experiments, we notice that coherence plays an important role in poetry generation (λ=10), while inter-line information is essential to depict the content coherence (μ=0.2). In the future, we plan to incorporate more criteria into the constrained summarization framework, e.g., maintenance of *parallelism* across lines, Part-of-Speech tagging and positive/negative sentimental labeling.

Acknowledgments

We thank all the anonymous reviewers for their useful feedbacks. This work was partially supported by National Sci-

ence Council, National Taiwan University and Intel Corporation under Grants NSC101-2911-I-002-001, NSC101-2628-E-002-028-MY2 and NTU102R7501, and by NSFC under Grants 61272340, 61050009, 61271304, and Key Program of Beijing Municipal Natural Science Foundation under Grants KZ201311232037. Rui Yan would like to thank Chia-Jung Lee for her inspiration.

References

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[Greene *et al.*, 2010] Erica Greene, Tugba Bodrumlu, and Kevin Knight. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pages 524–533, 2010.

[He *et al.*, 2012] J. He, M. Zhou, and L. Jiang. Generating chinese classical poems with statistical machine translation models. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[Hofmann, 2001] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42:177–196, 2001. 10.1023/A:1007617005950.

[Jiang and Zhou, 2008] Long Jiang and Ming Zhou. Generating chinese couplets using a statistical mt approach. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 377–384, 2008.

[Landauer *et al.*, 1998] T.K. Landauer, P.W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[Lin and Hovy, 2003] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 71–78, 2003.

[Liu, 1735] Wen-Wei Liu. The essense of poetry. 1735.

[Manurung *et al.*, 2011] R. Manurung, G. Ritchie, and H. Thompson. Using genetic algorithms to create meaningful poetic text. *Journal of Experimental & Theoretical Artificial Intelligence*, 24(1):43–64, 2011.

[Manurung, 2004] H. Manurung. An evolutionary algorithm approach to poetry generation. *University of Edinburgh. College of Science and Engineering. School of Informatics.*, 2004.

[Neto *et al.*, 2000] J.L. Neto, A.D. Santos, C.A.A. Kaestner, D. Santos, et al. Document clustering and text summarization. 2000.

[Netzer *et al.*, 2009] Yael Netzer, David Gabay, Yoav Goldberg, and Michael Elhadad. Gaiku: generating haiku with word associations norms. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, CALC '09, pages 32–39, 2009.

[Oliveira, 2009] H. Oliveira. Automatic generation of poetry: an overview. *Universidade de Coimbra*, 2009.

[Oliveira, 2012] H.G. Oliveira. Poetryme: a versatile platform for poetry generation. *Computational Creativity, Concept Invention, and General Intelligence*, 1:21, 2012.

[Radev *et al.*, 2004] D.R. Radev, H. Jing, and M. Sty. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40(6):919–938, 2004.

[Tosa *et al.*, 2008] N. Tosa, H. Obara, and M. Minoh. Hitch haiku: An interactive supporting system for composing haiku poem. *Entertainment Computing-ICEC 2008*, pages 209–216, 2008.

[Wan and Yang, 2008] Xiaojun Wan and Jianwu Yang. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 299–306, 2008.

[Wang, 2002] Li Wang. A summary of rhyming constraints of chinese poems. Beijing Press, 2002.

[Wu *et al.*, 2009] X. Wu, N. Tosa, and R. Nakatsu. New hitch haiku: An interactive renku poem composition supporting tool applied for sightseeing navigation system. *Entertainment Computing–ICEC 2009*, pages 191–196, 2009.

[Yan *et al.*, 2011a] Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 433–443, 2011.

[Yan *et al.*, 2011b] Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 745–754, 2011.

[Zhou *et al.*, 2010] Cheng-Le Zhou, Wei You, and Xiaojun Ding. Genetic algorithm and its implementation of automatic generation of chinese songci. *Journal of Software*, 21(3):427–437, 2010.